Mitglied der Helmholtz-Gemeinschaft

HPC–LEAP
EUROPEAN JOINT DOCTORATES

JÜLICH
FORSCHUNGSZENTRUM

HPC Training within the HPC-LEAP Program for European Joint Doctorates

D. Pleiter | Workshop on HPC Training Best Practices | 14 November 2016

HPC-LEAP Overview

Marie-Curie Training Network funded by European Union

- 17 partners
- Universities, research labs, commercial operators

Goal

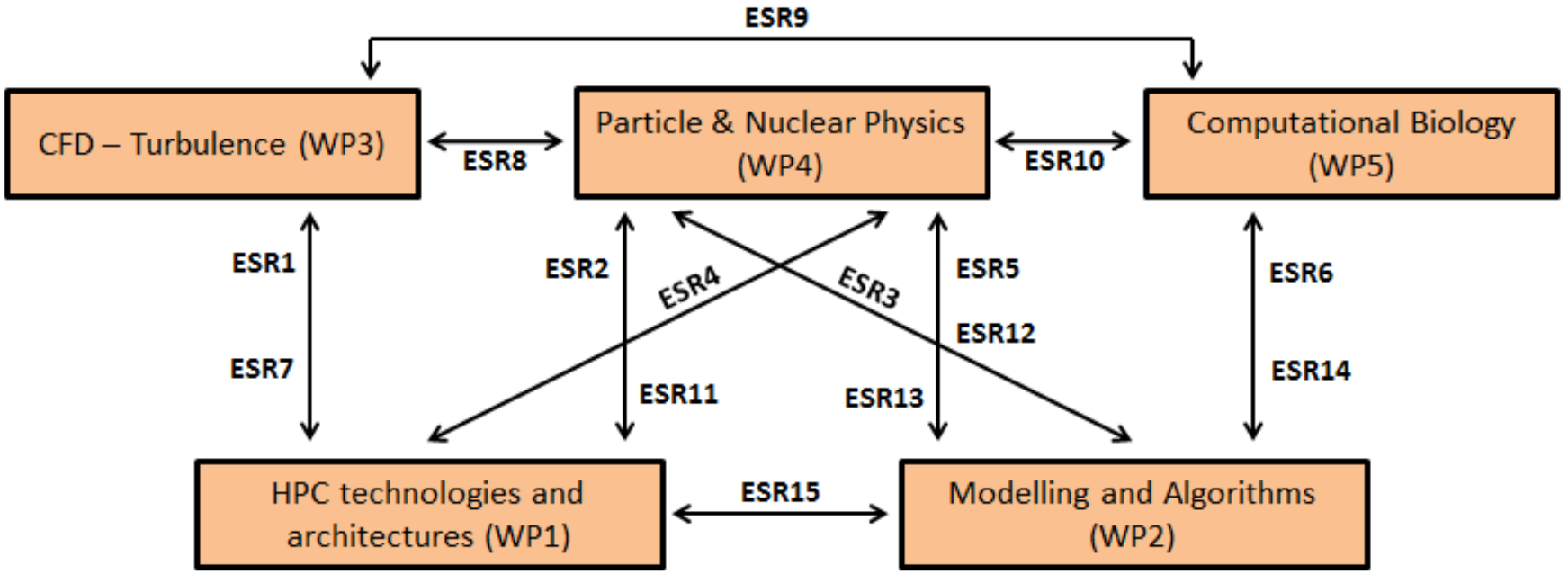
- Educate the next generation of scientists to address exascale computing challenges and enabling them to be at the forefront of their respective research fields

Highly interdisciplinary approach

- Multiple computational sciences disciplines
- Modelling and algorithms
- HPC architectures and technologies

15 Early Stage Researcher projects

HPC-LEAP Overview (cont.)



HPC Educational Goals

Provide knowledge about modern HPC architectures

- Starting from computer architecture principles

Improve skills to program such architectures

- Focus on parallel programming

Create understanding of attainable performance

- Lay basis for educated choice of optimal implementation
  - Including choice of algorithms
- Performance modelling as a tool

Practice performance analysis

- Need skill to measure performance

Improve performance optimisation skills

- Apply knowledge

HPC Winter School 2016

Organisation

- 3 weeks, 16 participants
- Lectures + exercises, keynotes, student projects

Lecture topics

- Computer architectures and performance modelling
- Performance analysis and optimisation
- Parallel algorithms
- MPI and OpenMP, GPU programming
- Mathematical libraries, parallel I/O, visualisations

Student projects

- 5 projects
- 3-4 students + 1 tutor per project
- 2 * 90 + 2 * 180 minute tutorial sessions
- Vanilla, serial code provided

Student Projects: Topics

Comparative analysis STREAM on Xeon, POWER, ARM

- Comparison of achievable memory bandwidth

Parallel matrix-matrix multiply

- Analysis of different parallel algorithms

Many-body code on GPU

- Practice GPU programming

Parallelisation of Lattice Boltzmann Application

- Multi-level parallelisation of given code

Parallelisation and optimisation of code simulating SU(3) Yang-Mills theory

- Multi-level parallelisation of given code

Student Projects: Expectations

Architectural analysis

- What are the features of the architecture that are relevant?

Exploration of implementation options and implementation

- What implementation options are available?
- Test implementations
- Final implementation and test for correctness

Performance analysis

- What is the actually obtained resource utilization?

Performance modelling

- What performance could I hope to achieve?

Presentation of results

Feedback on Student Projects

Questionnaire

	Neutral	Agree	Strongly agree
The projects have been relevant for my work?		55%	45%
How do you rate the support for the projects?	27%	36%	36%
The time allocated for working on the projects was too little?	36%	18%	45%

Specific comments

- Consider either allocating more time for working on projects or shrinking the projects dimension (which in my opinion would be a pity since the final outcome was pretty nice)
- The projects were really useful and more time allocated in the schedule would be an improvement.

Conclusions

Interdisciplinary training important as well as successful

- Computational sciences vs. algorithms vs. computer sciences

Need to combine lectures/keynotes and practical elements

- Exercises
- Student projects

Participants responded well to student project challenge

- Main issue: Allocated amount of time